PRESENTATION WEB DATAMINING PROJECT

LUCAS LE LORIER

AIMA MOHAMMAD

JONATHAN NGALAMULUME

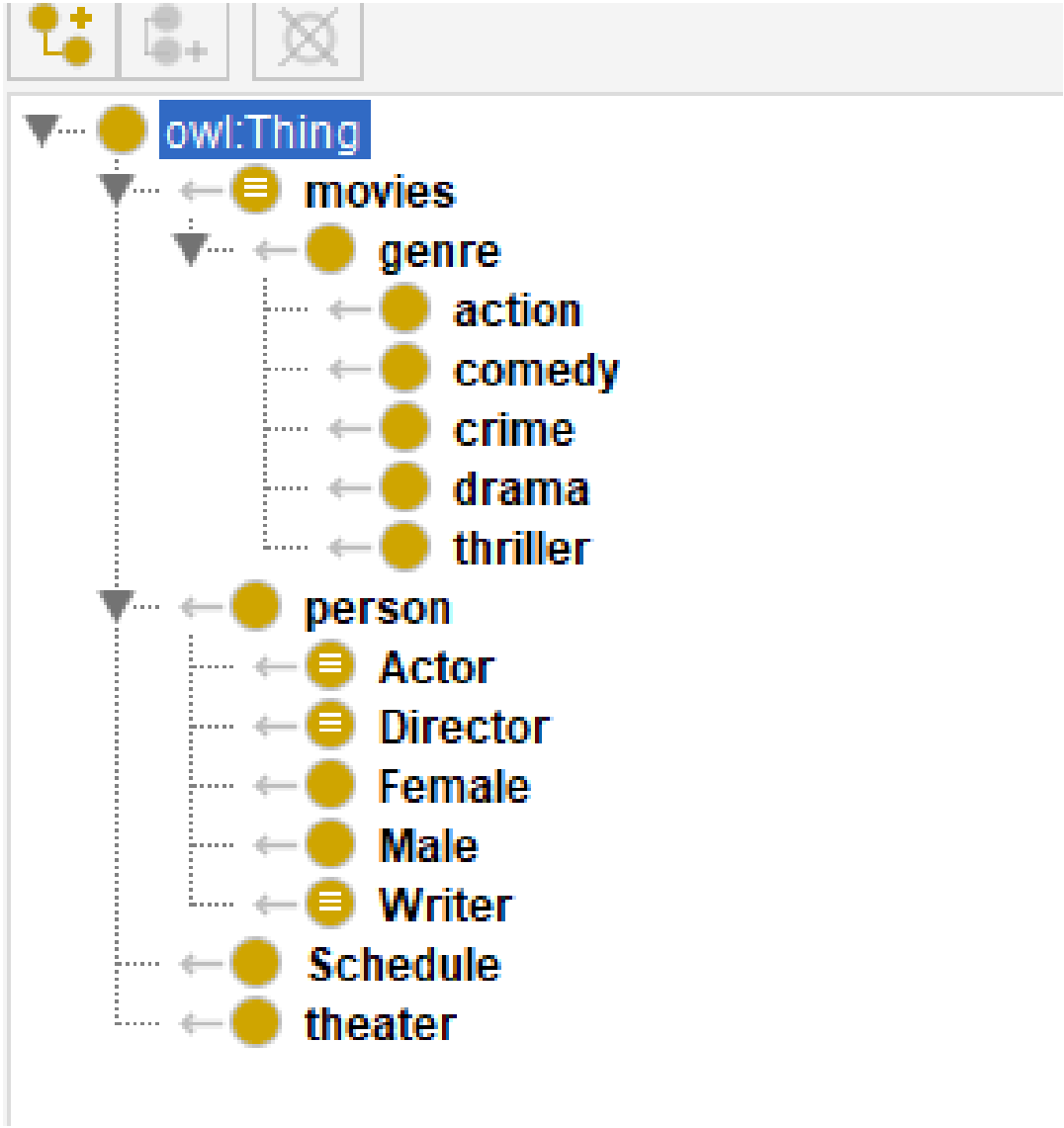
DIA4

PART I&II : MODELING MOVIES.OWL AND CREATING ITS DATABASE

DESIGN OF CLASSES AND SUBCLASSES OF MOVIES ONTOLOGY

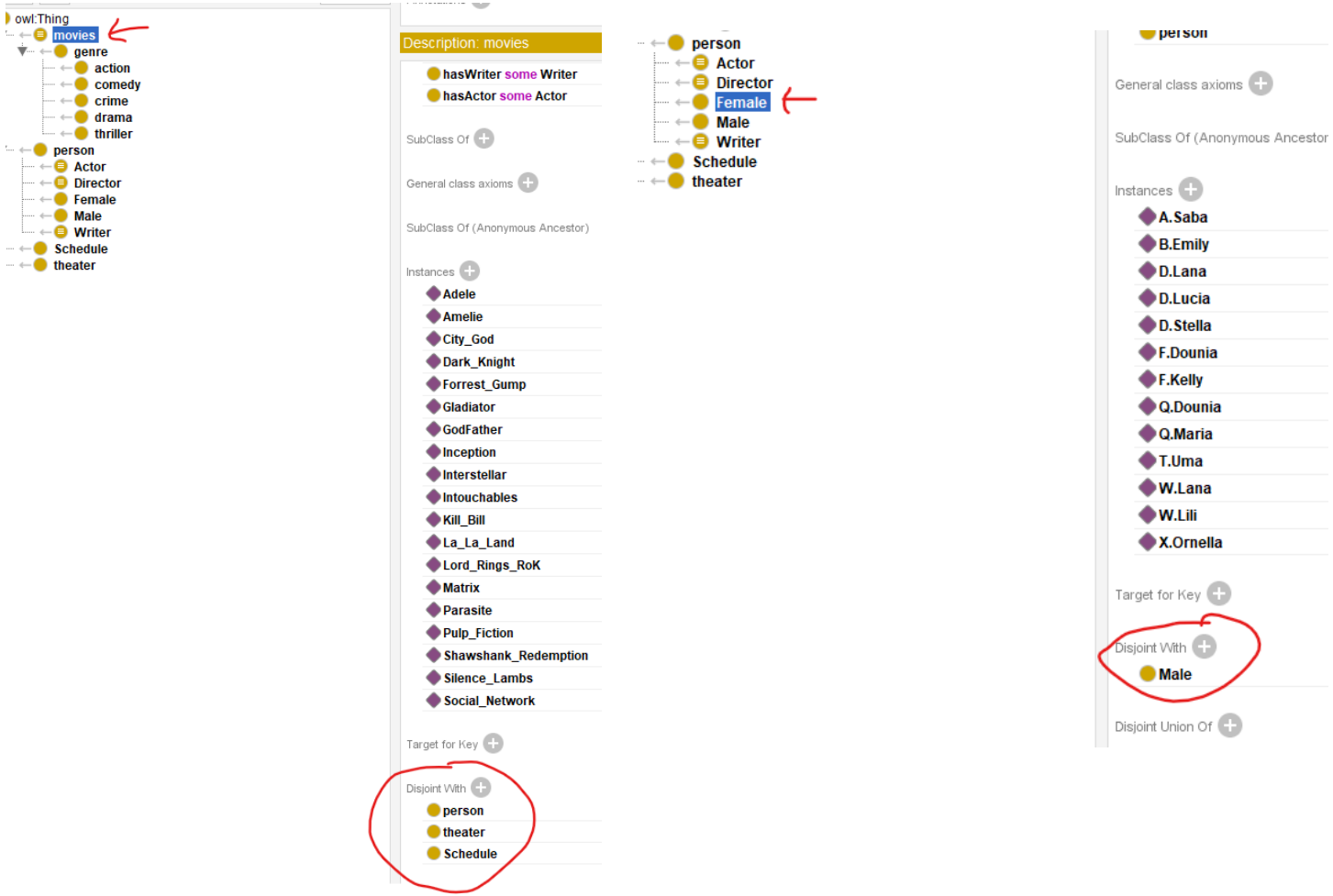
Choices :

- The genres are subclasses of the class movies rather than being its attributes.

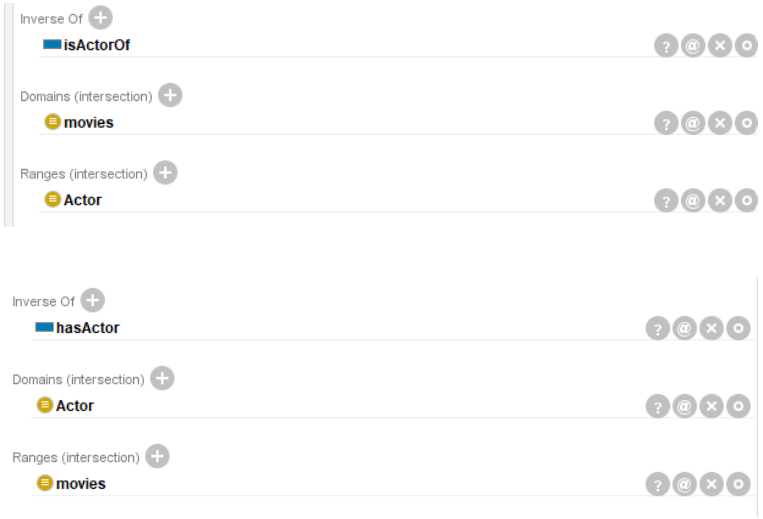


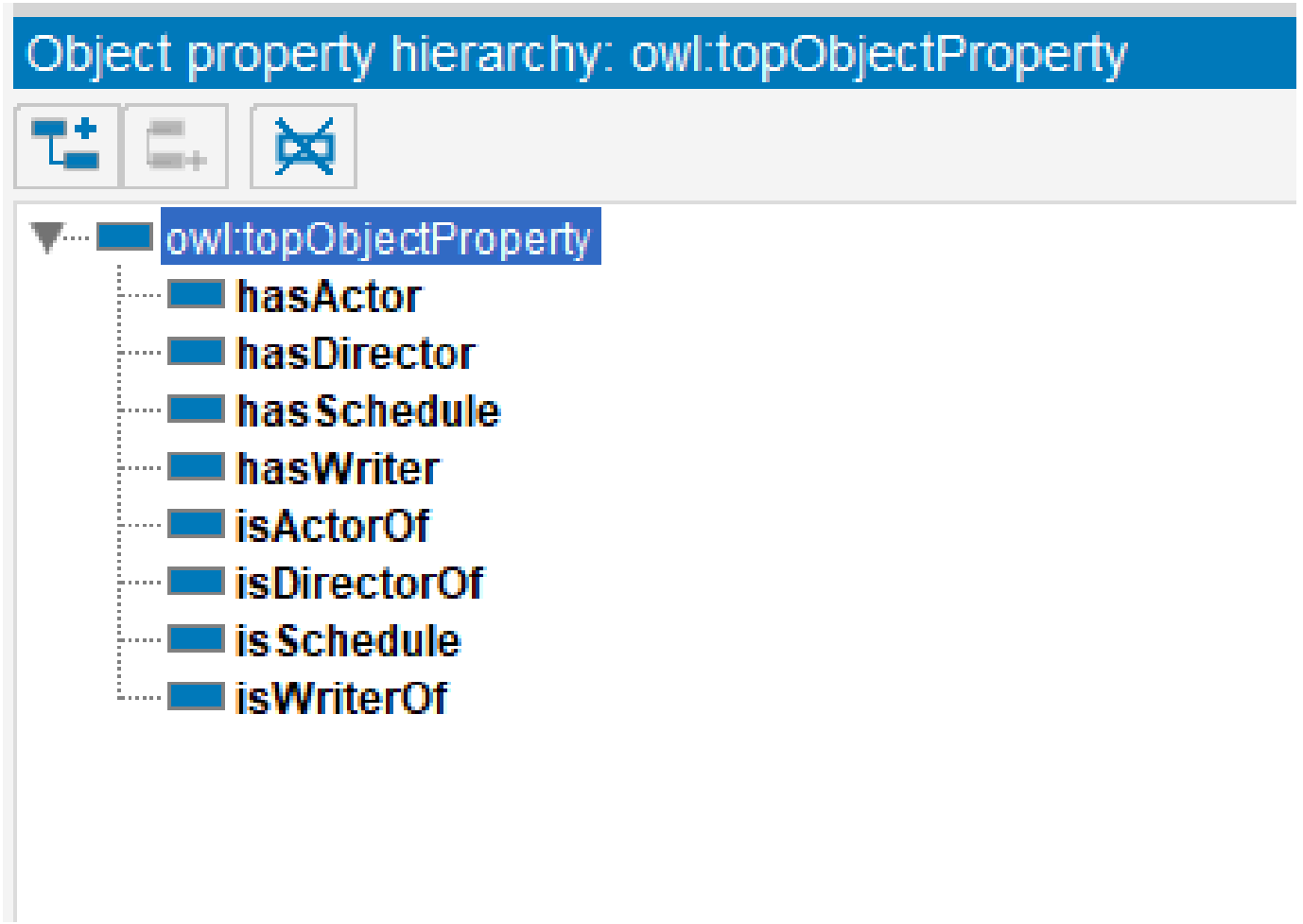
RESTRICTIONS ON CLASSES

- The classes Schedule, Person, Movies and Theater are disjoint.
- The classes Male and Female are disjoint.

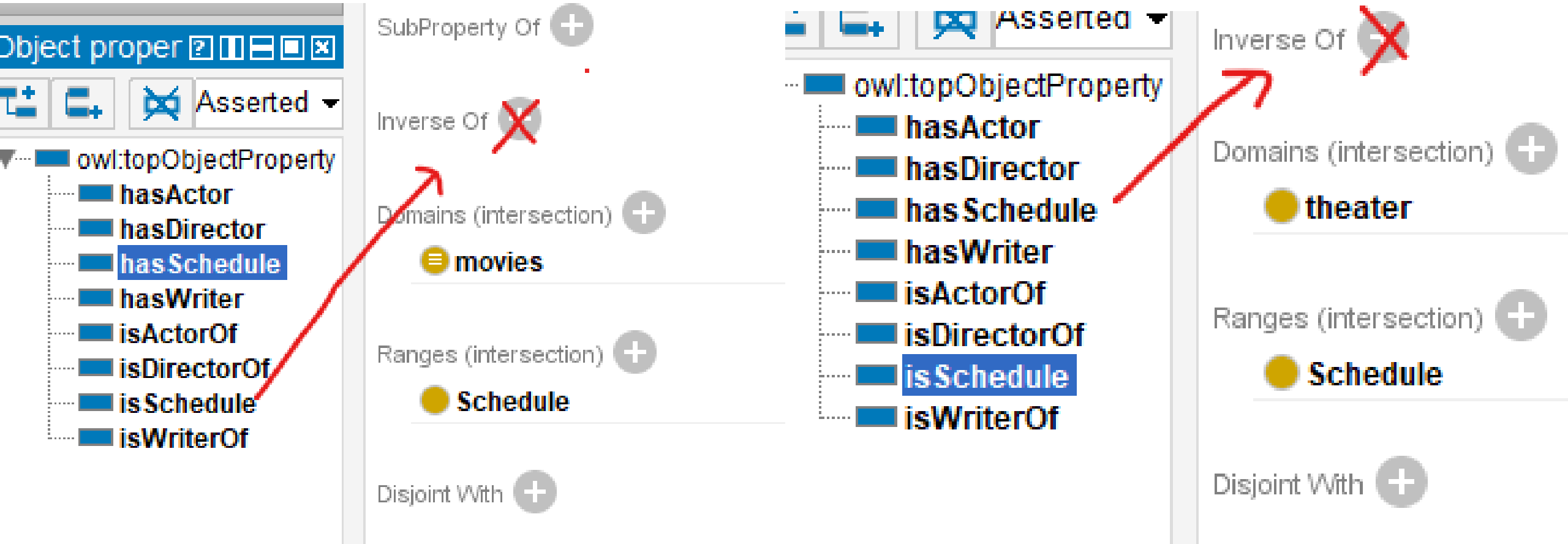


DESIGN OF CLASS PROPERTIES



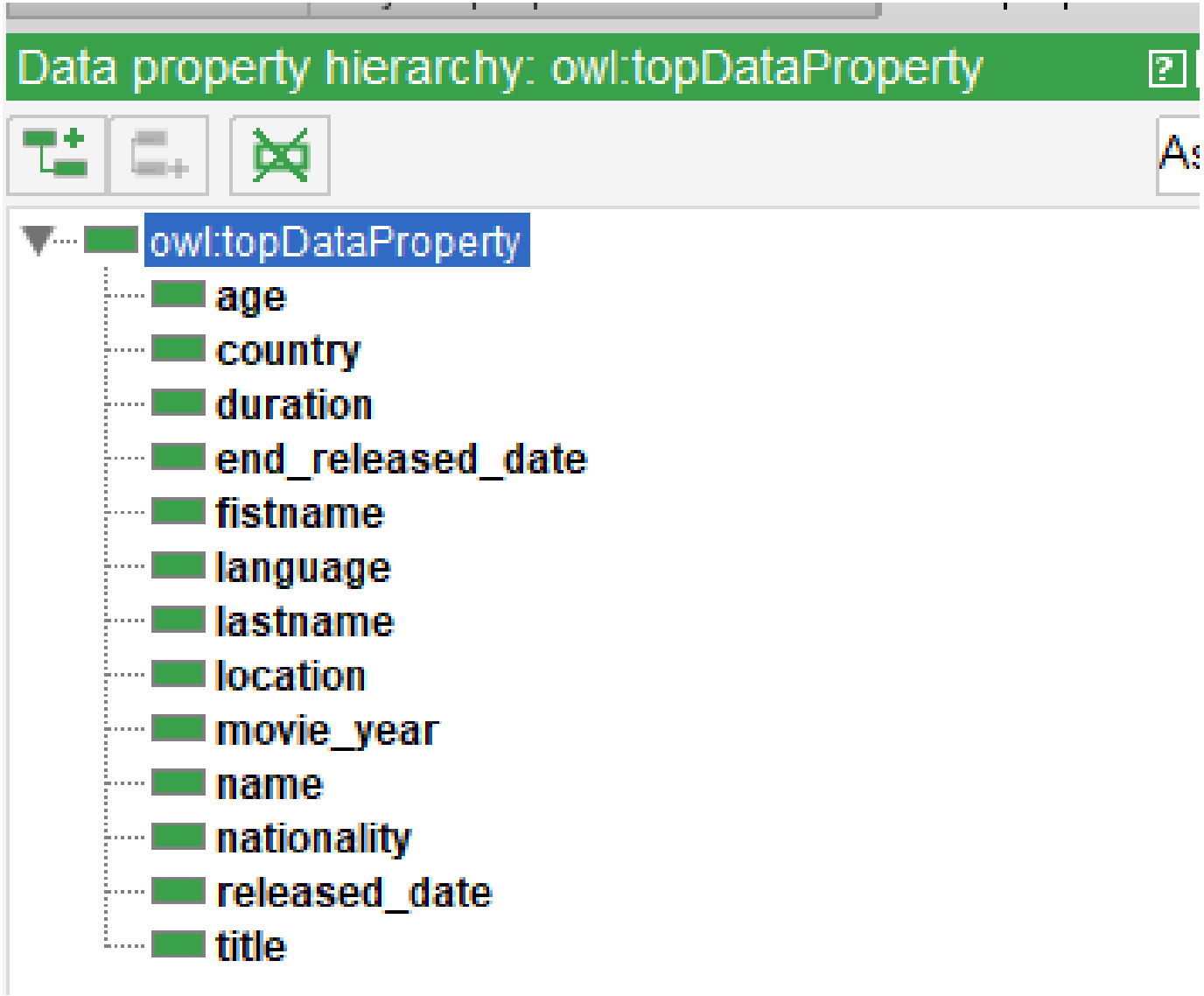


DESIGN OF CLASS PROPERTIES: THE SCHEDULE PROPERTY

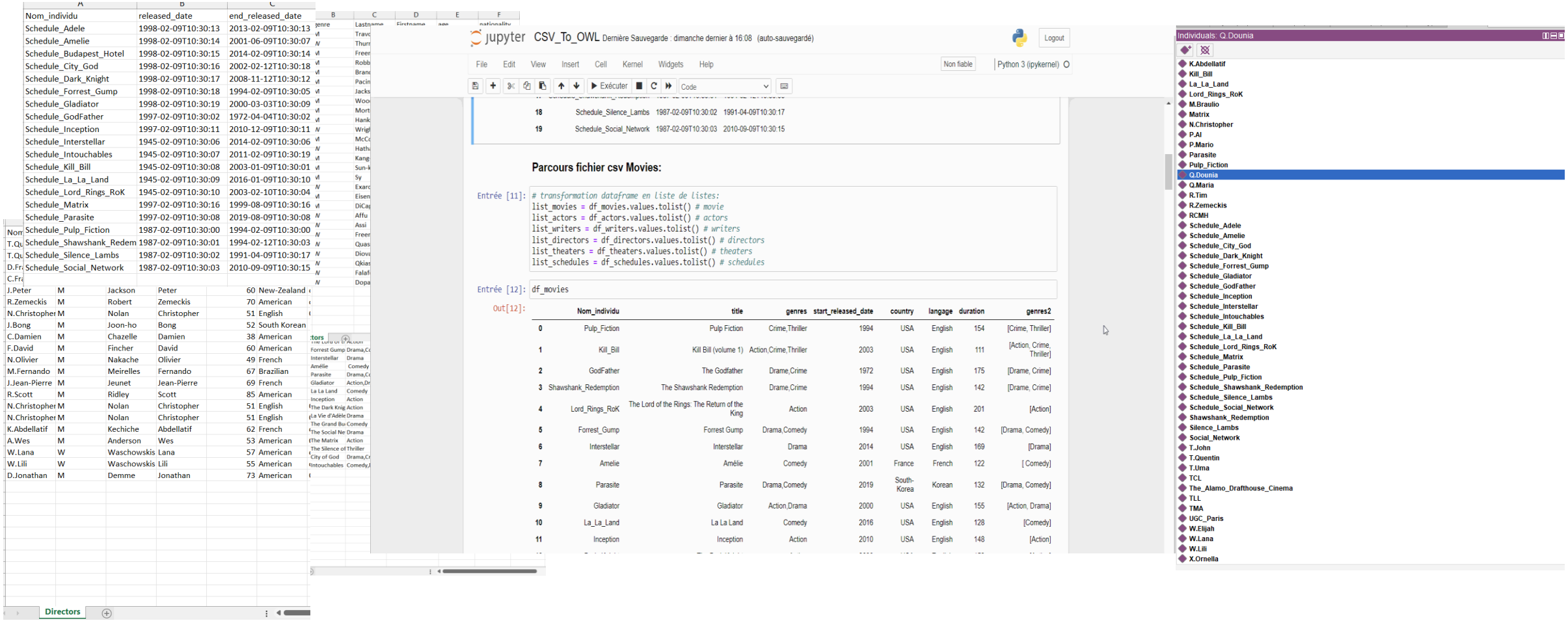


DATA PROPERTIES AND THEIR RANGES

- Age/duration/movie_year are positiveInteger.
- Released_date/end_released_date are dateTime.
- The others are of type string.



ADDING INDIVIDUALS -DATABASE PROTEGE



Nom_individu	released_date	end_released_date
Schedule_Adele	1998-02-09T10:30:13	2013-02-09T10:30:13
Schedule_Amelie	1998-02-09T10:30:14	2001-06-09T10:30:07
Schedule_Budapest_Hotel	1998-02-09T10:30:15	2014-02-09T10:30:14
Schedule_City_God	1998-02-09T10:30:16	2002-02-12T10:30:18
Schedule_Dark_Knight	1998-02-09T10:30:17	2008-11-12T10:30:12
Schedule_Forrest_Gump	1998-02-09T10:30:18	1994-02-09T10:30:05
Schedule_Gladiator	1998-02-09T10:30:19	2000-03-03T10:30:09
Schedule_GodFather	1997-02-09T10:30:02	1972-04-04T10:30:02
Schedule_Inception	1997-02-09T10:30:11	2010-12-09T10:30:11
Schedule_Interstellar	1945-02-09T10:30:06	2014-02-09T10:30:06
Schedule_Intouchables	1945-02-09T10:30:07	2011-02-09T10:30:19
Schedule_Kill_Bill	1945-02-09T10:30:08	2003-01-09T10:30:01
Schedule_La_La_Land	1945-02-09T10:30:09	2016-01-09T10:30:10
Schedule_Lord_Rings_RoK	1945-02-09T10:30:10	2003-02-10T10:30:04
Schedule_Matrix	1997-02-09T10:30:16	1999-08-09T10:30:16
Schedule_Parasite	1997-02-09T10:30:08	2019-08-09T10:30:08
Schedule_Pulp_Fiction	1987-02-09T10:30:00	1994-02-09T10:30:00
Schedule_Shawshank_Redem	1987-02-09T10:30:01	1994-02-12T10:30:03
Schedule_Silence_Lambs	1987-02-09T10:30:02	1991-04-09T10:30:17
Schedule_Social_Network	1987-02-09T10:30:03	2010-09-09T10:30:15

J.Peter	M	Jackson	Peter	60	New-Zealand
R.Zemeckis	M	Robert	Zemeckis	70	American
N.Christopher	M	Nolan	Christopher	51	English
J.Bong	M	Joon-ho	Bong	52	South Korean
C.Damien	M	Chazelle	Damien	38	American
F.David	M	Fincher	David	60	American
N.Olivier	M	Nakache	Olivier	49	French
M.Fernando	M	Meirelles	Fernando	67	Brazilian
J.Jean-Pierre	M	Jeunet	Jean-Pierre	69	French
R.Scott	M	Ridley	Scott	85	American
N.Christopher	M	Nolan	Christopher	51	English
N.Christopher	M	Nolan	Christopher	51	English
K.Abdellatif	M	Kechiche	Abdellatif	62	French
A.Wes	M	Anderson	Wes	53	American
W.Lana	W	Waschowskis	Lana	57	American
W.Lili	W	Waschowskis	Lili	55	American
D.Jonathan	M	Demme	Jonathan	73	American

Directors

jupyter CSV_To_OWL Dernière Sauvegarde : dimanche dernier à 16:08 (auto-sauvegardé)

Logout

File Edit View Insert Cell Kernel Widgets Help

Non fiable | Python 3 (ipykernel)

Exécuter | Code

18	Schedule_Silence_Lambs	1987-02-09T10:30:02	1991-04-09T10:30:17
19	Schedule_Social_Network	1987-02-09T10:30:03	2010-09-09T10:30:15

Parcours fichier csv Movies:

Entrée [11]:
```
# transformation dataframe en liste de listes:
list_movies = df_movies.values.tolist() # movie
list_actors = df_actors.values.tolist() # actors
list_writers = df_writers.values.tolist() # writers
list_directors = df_directors.values.tolist() # directors
list_theaters = df_theaters.values.tolist() # theaters
list_schedules = df_schedules.values.tolist() # schedules
```

Entrée [12]:
```
df_movies
```

Out[12]:

	Nom_individu	title	genres	start_released_date	country	langage	duration	genres2
0	Pulp_Fiction	Pulp Fiction	Crime,Thriller	1994	USA	English	154	[Crime, Thriller]
1	Kill_Bill	Kill Bill (volume 1)	Action,Crime,Thriller	2003	USA	English	111	[Action, Crime, Thriller]
2	GodFather	The Godfather	Drame,Crime	1972	USA	English	175	[Drame, Crime]
3	Shawshank_Redemption	The Shawshank Redemption	Drame,Crime	1994	USA	English	142	[Drame, Crime]
4	Lord_Rings_RoK	The Lord of the Rings: The Return of the King	Action	2003	USA	English	201	[Action]
5	Forrest_Gump	Forrest Gump	Drama,Comedy	1994	USA	English	142	[Drama, Comedy]
6	Interstellar	Interstellar	Drama	2014	USA	English	169	[Drama]
7	Amelie	Amélie	Comedy	2001	France	French	122	[Comedy]
8	Parasite	Parasite	Drama,Comedy	2019	South-Korea	Korean	132	[Drama, Comedy]
9	Gladiator	Gladiator	Action,Drama	2000	USA	English	155	[Action, Drama]
10	La_La_Land	La La Land	Comedy	2016	USA	English	128	[Comedy]
11	Inception	Inception	Action	2010	USA	English	148	[Action]

Individuals: Q.Dounia

- K.Abdellatif
- Kill_Bill
- La_La_Land
- Lord_Rings_RoK
- M.Braulio
- Matrix
- N.Christopher
- P.Al
- P.Mario
- Parasite
- Pulp_Fiction
- Q.Dounia
- Q.Maria
- R.Tim
- R.Zemeckis
- RCMH
- Schedule_Adele
- Schedule_Amelie
- Schedule_City_God
- Schedule_Dark_Knight
- Schedule_Forrest_Gump
- Schedule_Gladiator
- Schedule_GodFather
- Schedule_Inception
- Schedule_Interstellar
- Schedule_Intouchables
- Schedule_Kill_Bill
- Schedule_La_La_Land
- Schedule_Lord_Rings_RoK
- Schedule_Matrix
- Schedule_Parasite
- Schedule_Pulp_Fiction
- Schedule_Shawshank_Redemption
- Schedule_Silence_Lambs
- Schedule_Social_Network
- Shawshank_Redemption
- Silence_Lambs
- Social_Network
- T.John
- T.Quentin
- T.Uma
- TCL
- The_Alamo_Drafthouse_Cinema
- TLL
- TMA
- UGC_Paris
- W.Elijah
- W.Lana
- W.Lili
- X.Ornella

PART III : SPARQL QUERIES ON MOVIES.OWL

List the instances of the class Actor

Active ontology × | Entities × | Individuals by class × | DL Query × | SPARQL Query ×

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

SELECT DISTINCT ?prenom ?nom
 WHERE{?Actor this:isActorOf ?Movie.
?Actor this:fistname ?prenom.
?Actor this:lastname ?nom.
}
```

prenom	nom
"Stella"^^<http://www.w3.org/2001/XMLSchema#string>	"Dopa"^^<http://www.w3.org/2001/XMLSchema#string>
"Dounia"^^<http://www.w3.org/2001/XMLSchema#string>	"Falak"^^<http://www.w3.org/2001/XMLSchema#string>
"Uma"^^<http://www.w3.org/2001/XMLSchema#string>	"Thurman"^^<http://www.w3.org/2001/XMLSchema#string>
"Samuel L."^^<http://www.w3.org/2001/XMLSchema#string>	"Jackson"^^<http://www.w3.org/2001/XMLSchema#string>
"Dounia"^^<http://www.w3.org/2001/XMLSchema#string>	"Quento"^^<http://www.w3.org/2001/XMLSchema#string>
"Tim"^^<http://www.w3.org/2001/XMLSchema#string>	"Robbins"^^<http://www.w3.org/2001/XMLSchema#string>

List the name of all Thriller movies. For each one, display its director.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

SELECT DISTINCT ?Thriller_movie ?director

WHERE{

?Thriller_movie rdf:type this:thriller.

?director this:isDirectorOf ?Thriller_movie.

}
```

Thriller_movie	director
Kill_Bill	A.Wes
Silence_Lambs	J.Peter
Pulp_Fiction	T.Quentin

List the name of all Crime Thriller movies.

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

SELECT DISTINCT ?Thriller_Crime_movie
 WHERE{

?Thriller_Crime_movie rdf:type this:thriller.
?Thriller_Crime_movie rdf:type this:crime.

}
```

Thriller_Crime_movie
Kill_Bill
Pulp_Fiction

List the name of Actors older than 51 years.

File Edit View Reasoner Tools Refactor Window Help

< > untitled-ontology-11 (http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11) Search...

Active ontology × | Entities × | Individuals by class × | DL Query × | SPARQL Query ×

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>


SELECT DISTINCT ?prenom ?nom ?age
WHERE{
?Actor this:isActorOf ?Movie.
?Actor this:age ?age.
?Actor this:fistname ?prenom.
?Actor this:lastname ?nom.
FILTER(?age>"51"^^xsd:positiveInteger).
}
```

prenom	nom	age
"Samuel L."^^<http://www.w3.org/2001/XMLSchema#string>	"Jackson"^^<http://www.w3.org/2001/XMLSchema#string>	"73"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"Dounia"^^<http://www.w3.org/2001/XMLSchema#string>	"Quento"^^<http://www.w3.org/2001/XMLSchema#string>	"56"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"Tim"^^<http://www.w3.org/2001/XMLSchema#string>	"Robbins"^^<http://www.w3.org/2001/XMLSchema#string>	"63"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"Morgan"^^<http://www.w3.org/2001/XMLSchema#string>	"Freeman"^^<http://www.w3.org/2001/XMLSchema#string>	"84"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"Al"^^<http://www.w3.org/2001/XMLSchema#string>	"Pacino"^^<http://www.w3.org/2001/XMLSchema#string>	"82"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"Lana"^^<http://www.w3.org/2001/XMLSchema#string>	"Danom"^^<http://www.w3.org/2001/XMLSchema#string>	"86"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"Frank"^^<http://www.w3.org/2001/XMLSchema#string>	"Darabont"^^<http://www.w3.org/2001/XMLSchema#string>	"53"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
"John"^^<http://www.w3.org/2001/XMLSchema#string>	"Travolta"^^<http://www.w3.org/2001/XMLSchema#string>	"59"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>

List of movies that are played in theater for a specific day and where and until when

Active ontology × | Entities × | Individuals by class × | DL Query × | SPARQL Query ×

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

SELECT ?movie ?date_fin ?cinema WHERE
{
?movie rdf:type this:movies.
?movie this:hasSchedule ?x.
?x this:end_released_date ?date_fin.
?x  this:released_date ?d.
FILTER((?d<"1998-02-09T10:30:13"^^xsd:dateTime)&&(?date_fin>"1998-02-09T10:30:13"^^xsd:dateTime)).
?cinema this:isSchedule ?x.

}
```

movie	date_fin	cinema
Matrix	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TMA
Parasite	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TMA
Shawshank_Redemption	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TMA
La_La_Land	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TLL
La_La_Land	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	UGC_Paris
Kill_Bill	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TLL
Silence_Lambs	"2004-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TMA
Dark_Knight	"2013-02-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	The_Alamo_Drafthouse_Cinema
Lord_Rings_RoK	"2014-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	RCMH
Inception	"2003-02-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TCL
Inception	"2003-02-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	UGC_Paris
Forrest_Gump	"2013-02-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	UGC_Paris
Forrest_Gump	"2013-02-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TCL
Social_Network	"2004-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	TMA
Social_Network	"2004-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	UGC_Paris
Social_Network	"2004-09-09T10:30:13"^^<http://www.w3.org/2001/XMLScl	RCMH

A query that contains at least 2 Optional Graph Patterns

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

SELECT DISTINCT ?Movie ?director ?duration  WHERE{

?Movie rdf:type this:movies.


OPTIONAL{ ?director this:isDirectorOf  ?Movie.}



OPTIONAL{ ?Movie this:duration ?duration.

FILTER (?duration < "140"^^xsd:positiveInteger)}



}
```

Movie	director	duration
Gladiator	D.Frank	
Forrest_Gump	J.Peter	
GodFather	A.Wes	
Matrix	J.Peter	"136"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
City_God	T.Quentin	"130"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
Intouchables	T.Quentin	"112"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
Parasite	C.FrancisF	"132"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
Silence_Lambs	J.Peter	"118"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
Social_Network	W.Lili	
Interstellar	W.Lili	
Interstellar	W.Lana	
Interstellar	C.FrancisF	
Shawshank_Redemption	N.Christopher	
Shawshank_Redemption	A.Wes	
Adele	W.Lili	
Dark_Knight	D.Jonathan	
Dark_Knight	T.Uma	
Dark_Knight	T.Quentin	
Inception	R.Zemeckis	
Inception	D.Jonathan	
Pulp_Fiction	T.Quentin	
Kill_Bill	A.Wes	"111"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>
La_La_Land	D.Frank	"128"^^<http://www.w3.org/2001/XMLSchema#positiveInteger>

A query that contains at least 2 alternatives and conjunctions

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

SELECT ?nom ?prenom ?age WHERE

{{?Actor this:isActorOf ?Movie.
?Actor this:age ?age.
FILTER (?age>="18"^^xsd:positiveInteger && ?age<="20"^^xsd:positiveInteger)}
UNION
{?Actor this:age ?age.
FILTER (?age>="40"^^xsd:positiveInteger && ?age<="55"^^xsd:positiveInteger)}
?Actor this:fistname ?prenom.
?Actor this:lastname ?nom.}
```

nom	prenom	age
"Anderson"^^<http://www.w3.org/2001/XMLSchema#string	"Wes"^^<http://www.w3.org/2001/XMLSchema#string>	"53"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Thurman"^^<http://www.w3.org/2001/XMLSchema#string	"Uma"^^<http://www.w3.org/2001/XMLSchema#string>	"43"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Washowsky"^^<http://www.w3.org/2001/XMLSchema#stri	"Lili"^^<http://www.w3.org/2001/XMLSchema#string>	"55"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Tarantino"^^<http://www.w3.org/2001/XMLSchema#string	"Quentin"^^<http://www.w3.org/2001/XMLSchema#string>	"53"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Danko"^^<http://www.w3.org/2001/XMLSchema#string>	"Lucia"^^<http://www.w3.org/2001/XMLSchema#string>	"46"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Xiao"^^<http://www.w3.org/2001/XMLSchema#string>	"Ornella"^^<http://www.w3.org/2001/XMLSchema#string>	"45"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Darabont"^^<http://www.w3.org/2001/XMLSchema#string	"Frank"^^<http://www.w3.org/2001/XMLSchema#string>	"53"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Nolan"^^<http://www.w3.org/2001/XMLSchema#string>	"Christopher"^^<http://www.w3.org/2001/XMLSchema#stri	"51"^^<http://www.w3.org/2001/XMLSchema#positiveInte
"Wood"^^<http://www.w3.org/2001/XMLSchema#string>	"Elijah"^^<http://www.w3.org/2001/XMLSchema#string>	"41"^^<http://www.w3.org/2001/XMLSchema#positiveInte

A query that contains a CONSTRUCT query form

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

CONSTRUCT {
?movie this:title ?title.
?movie this:director ?director.
?movie this:duration ?duration.}
WHERE
{?movie rdf:type this:movies.
?movie this:title ?title.
?director this:isDirectorOf ?movie.
?movie this:duration ?duration.}
```

Subject	Predicate	Object
Matrix	title	"The Matrix"^^<http://www.w3.org/2001/XMLSchema#strin
Matrix	director	J.Peter
Matrix	duration	"136"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Shawshank_Redemption	title	"The Shawshank Redemption"^^<http://www.w3.org/200
Shawshank_Redemption	director	N.Christopher
Shawshank_Redemption	duration	"142"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Shawshank_Redemption	director	A.Wes
Parasite	title	"Parasite"^^<http://www.w3.org/2001/XMLSchema#string
Parasite	director	C.FrancisF
Parasite	duration	"132"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Dark_Knight	title	"The Dark Knight"^^<http://www.w3.org/2001/XMLSchem
Dark_Knight	director	D.Jonathan
Dark_Knight	duration	"152"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Dark_Knight	director	T.Uma
Dark_Knight	director	T.Quentin
Adele	title	"La Vie d'Adele"^^<http://www.w3.org/2001/XMLSchema#
Adele	director	W.Lili
Adele	duration	"160"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Interstellar	title	"Interstellar"^^<http://www.w3.org/2001/XMLSchema#stri
Interstellar	director	W.Lili
Interstellar	duration	"169"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Interstellar	director	W.Lana
Interstellar	director	C.FrancisF

Execute

Votre écran est partagé par le biais de l'application

A query that contains an ASK query form

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

ASK  WHERE{
?Actor this:isActorOf ?Movie.

?Actor this:fistname "Morgan"^^xsd:string.
?Actor this:lastname "Freeman"^^xsd:string.


}
```

Result
True

A query that contains a DESCRIBE query form

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX this:<http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#>

DESCRIBE ?movie_comedy

WHERE {

  ?movie_comedy rdf:type this:comedy.

}
```

Subject	Predicate	Object
Parasite	rdf:type	comedy
Parasite	rdf:type	owl:NamedIndividual
Parasite	rdf:type	owl:NamedIndividual
Parasite	title	"Parasite"^^<http://www.w3.org/2001/XMLSchema#string
Parasite	rdf:type	owl:NamedIndividual
Parasite	rdf:type	owl:NamedIndividual
Parasite	rdf:type	movies
Parasite	rdf:type	owl:NamedIndividual
Parasite	language	"Korean"^^<http://www.w3.org/2001/XMLSchema#string>
Parasite	rdf:type	owl:NamedIndividual
Parasite	country	"South-Korea"^^<http://www.w3.org/2001/XMLSchema#st
Parasite	rdf:type	owl:NamedIndividual
Parasite	hasSchedule	Schedule_Parasite
Parasite	rdf:type	owl:NamedIndividual
Parasite	rdf:type	drama
Parasite	rdf:type	owl:NamedIndividual
Parasite	rdf:type	owl:NamedIndividual
Parasite	rdf:type	owl:NamedIndividual
Parasite	movie_year	"2019"^^<http://www.w3.org/2001/XMLSchema#positiveIn
Parasite	rdf:type	owl:NamedIndividual
Parasite	duration	"132"^^<http://www.w3.org/2001/XMLSchema#positiveInt
Parasite	rdf:type	owl:NamedIndividual
E Kelly	isWriterOf	Parasite

PART IV : MANIPULATING THE ONTOLOGY

LOAD OUR OWL MODEL INTO A GRAPH

```
Entrée [112]: g = Graph()
              g.parse("movies2.owl")

   Out[112]: <Graph identifier=N0b2deb6f13a84145a7de4919682529d0 (<class 'rdflib.graph.Graph'>)>
```

- 1.LOADS THE ONTOLOGY AND DISPLAYS ALL THE PERSONS (**WITHOUT** USING QUERIES, **WITHOUT** INFERENCE).

Loads the ontology and displays all the Persons (without using queries, without inference).

```
Entrée [104]: # Get the URI of the "Person" class
              person_uri = None
              for s, p, o in g:
                  if(p == URIRef("http://www.w3.org/1999/02/22-rdf-syntax-ns#type") and o == URIRef("http://www.semanticweb.org/mohammadaima/
                      print(s.split('#')[-1])
```

```
M.Braulio
J.Samuel_L.
P.Mario
N.Christopher
T.Uma
J.JPierre
J.Peter
D.Lucia
P.Al
W.Lili
B.Emily
R.Tim
T.Quentin
D.Lana
```

- 2.LOADS THE ONTOLOGY AND DISPLAYS ALL THE PERSONS (**USING** A QUERY, **WITHOUT** INFERENCE). CREATE THE USED QUERY IN TEXT FILE UNDER THE DATA FOLDER.

Entrée [109]:

```
# Define the namespaces
ns = {
    "owl": Namespace("http://www.w3.org/2002/07/owl#"),
    "rdf": Namespace("http://www.w3.org/1999/02/22-rdf-syntax-ns#"),
    "rdfs": Namespace("http://www.w3.org/2000/01/rdf-schema#"),
    "xsd" : Namespace("http://www.w3.org/2001/XMLSchema#"),
    "this": Namespace("http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#")
}

# Load the query from a file
with open("query.rq", "r") as f:
    query = f.read()

# Execute the query and print the results
results = g.query(query, initNs=ns)
for result in results:
    print(result)
```

```
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#A.Saba'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#A.Wes'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#B.Emily'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#B.Marlon'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#C.FrancisF'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#D.Frank'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#D.Jonathan'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#D.Lana'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#D.Lucia'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#D.Stella'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#F.Dounia'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#F.Kelly'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#F.Morgan'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#G.Winston'),)
(rdflib.term.URIRef('http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#J.JPierre'),)
```

- 3. LOADS THE ONTOLOGY AND DISPLAYS ALL THE ACTORS (**WITHOUT** USING QUERIES, **USING** INFERENCE).

3. Loads the ontology and displays all the Actors (without using queries, using inference).

Entrée [110]:

```
# Get the URI of the "Person" class
actors = set()
for s, p, o in g:
    if(p == URIRef("http://www.semanticweb.org/mohammadaima/ontologies/2023/2/untitled-ontology-11#isActorOf")):
    #if(p == URIRef("http://www.w3.org/1999/02/22-rdf-syntax-ns#isActorOf") and o == URIRef("http://www.semanticweb.org/mohammad
        actors.add(s)
for actor in actors:
    print(actor.split('#')[-1])
```

```
F.Morgan
B.Marlon
J.Samuel_L.
Q.Maria
T.John
P.Al
D.Lana
W.Elijah
D.Frank
Q.Dounia
R.Tim
T.Uma
F.Dounia
D.Stella
```

- 4. DEVELOPS A PROGRAM THAT :

- Reads a name of a movie
- If it doesn’t exist displays an error message
- Else, display its year, country, genres and actors
- Display their program where and when

Live demonstraton

THANK YOU ! :)